Advances in Automatically Solving the ENEM

Igor Cataneo Silveira
Institute of Mathematics and Statistics
Universidade de São Paulo
igorcs@ime.usp.br

Denis Deratani Mauá
Institute of Mathematics and Statistics
Universidade de São Paulo
ddm@ime.usp.br

***Abstract*—Answering questions formulated in natural language is a long standing quest in Artificial Intelligence. However, even formulating the problem in precise terms has proven to be too challenging, which lead many researchers to focus on Multiple-Choice Question Answering problems. One particularly interesting type of the latter problem is solving standardized tests such as university entrance exams. The Exame Nacional do Ensino Médio (ENEM) is a High School level exam widely used by Brazilian universities as entrance exam, and the world's second biggest university entrance examination in number of registered candidates. In this work we tackle the problem of answering purely textual multiple-choice questions from the ENEM. We build on a previous solution that formulated the problem as a text information retrieval problem. In particular, we investigate how to enhance these methods by text augmentation using Word Embedding and WordNet, a structured lexical database where words are connected according to some relations like synonymy and hypernymy. We also investigate how to boost performance by building ensembles of weakly correlated solvers. Our approaches obtain accuracies ranging from 26% to 29.3%, outperforming the previous approach.**

***Index Terms*—Multiple-Choice Question Answering, ENEM, Information Retrieval, Word Embedding**

I. Introduction

Automatically answering questions posed in natural language is an old quest of Artificial Intelligence research. This task, in its most generic form, consists of answering a question posed in free-text format by querying a knowledge base (KB), identifying relevant information, and producing a final answer in natural language. In adittion to the intrinsic difficulties of these subtasks, there are two major hindrances that challenge effectively designing Question Answering (QA) systems. First, solutions are restricted by their KB. Curated and structured KB are costly to build and maintain; consequently they are usually limited to narrow domains, while questions usually encompass much wider domains. Second, there is no agreement on how to assess the outputs of the system, or even to determine whether they constitute valid answers.

The Exame Nacional do Ensino Médio (ENEM) is a High-School level exam applied every year all over the country by the Brazilian government. In 2016, about 9.2 million people took the exam, only slightly less than the 9.4 million students that participated in the Chinese Gaokao exam, the world's largest university entrance exam in number of participants [1], [2]. The ENEM is used by many Brazilian universities as an entrance exam, and recently became part of an unified admission procedure followed by all Federal Public Universities in the country. The assessment comprises writing an essay about a general topic and answering a multiple-choice test involving four major areas: Humanities, Languages, Science and Mathematics. The questions are not segmented into subjects and many questions combine more than one scientific discipline, which makes it appealing as a QA benchmark. Importantly, exams and their solutions are publicly and freely available at a govern-hosted website (http://inep.gov.br/provas-e-gabaritos).

In a previous work, we proposed solving the Exame Nacional do Ensino Médio as a new and interesting benchmark for QA [3]. In fact, solving standardized tests such as university entrance exams is a type of Multiple-Choice Question Answering (MCQA), a simplified version of QA with two main advantages: it dispenses with the need of producing sentences in natural language, and it allows for automatic and objective evaluation of systems. In comparison with other university entrance exams used for MCQA [4], [5], the ENEM has a collection of freely available exam copies, has a wider collection of questions, is not topic-specific and promotes natural language processing for Brazilian Portuguese.

In this paper, we resume our initial work [3], and develop better techniques for answering questions from the ENEM using Text Information Retrieval. We focus on answering purely textual questions that do not rely on any form of image understanding or mathematical reasoning. That is, we ignore questions referring to diagrams, chemical or mathematical formulae, charts, drawings and pictures. We investigate three approaches. The first approach consists in augmenting the corpus or query by using WordNet [6], a handcrafted knowledge base of word similarities. The second approach uses state-of-the-art word embeddings learned from a large corpus of Brazilian Portuguese texts. The third approach consists in building ensembles of weakly correlated solvers obtained by the previous approaches. We investigate two variants: a weighted majority voting scheme where the set of solvers and their weights are selected by greedy search, and a an Support Vector Machine (SVM) classifier which takes the solvers output as input.

The rest of the paper is organized as follows. We start by reviewing related works on solving standardized tests in Section II. Then we review in Section III our previous work on solving the ENEM. Our contributions using text augmentation

The first author was supported by CAPES. The second author was partially supported by CNPq grants nos. 303920/2016-5 and 420669/2016-7, and FAPESP grant 2016/01055-1.

techniques appear in Section IV, and the techniques for producing ensembles are described in Section V. In Section VI we show the empirical results achieved. We conclude the paper in Section VII with a review of our motivations and contributions.

II. Related Work

Miyao and Kawazoe proposed using a Japanese university entrance exam as a less ambitious, but still challenging, benchmark for QA [4]. The proposed exam comprises 11 topics: Japanese, English, Mathematics, World History, Japanese History, Modern Society, Politics & Economics, Ethics, Physics, Chemistry and Biology. All the questions other than Mathematics are multiple-choice. The authors created a collection of MCQA problems by manually translating the exam's questions into a machine-readable format. In a 2014 competition, contestant systems were unable to match overall human performance; the best performance achieved was 58% in World History and 26% in English [7]. As a matter of fact, this dataset can be seen as 11 topic-specific datasets. This topic separation allows for using AI techniques to explore specific characteristics of a topic, thus not generalizing properly neither to other topics nor to other languages. Furthermore, this dataset is not publicly available.

A similar proposal was put forward by Cheng et al. [5], who addressed the problem of solving the *Gaokao*, a Chinese university entrance exam. This exam has three mandatory subjects (Chinese, Mathematics and Foreign Language), and three elective subjects: either Geography, History and Politics or Physics, Chemistry and Biology, according to the test-taker preference. The authors translated the questions into a machine-readable format and made available only the questions from the History subject, in which they report achieving 44% accuracy. Their approach relies in idiosyncrasies of the Chinese language and in specific characteristics of the exam. As with the Japanese benchmark, the segmentation into known topics makes this dataset topic-specific and with a relatively narrow domain.

In a previous work, we advocated the use of ENEM questions as a benchmark for Brazilian Portuguese MCQA [3]. To this end, we created a structured dataset of Humanities and Languages questions of the ENEM exams from 2010 to 2015. Each question was structured as a header, statement and five candidate answers. We labeled each question with knowledge tags indicating the type of knowledge one needs, in principle, to solve it: Text Comprehension (TC), Encyclopedic Knowledge (EK), Image Comprehension (IC), Domain Specific Knowledge (DS) and Mathematical Reasoning (MR). We also annotated questions with informative tags that described whether the question has an image in its body or a mention to chemical elements (CE) that cannot be treated simply as text. We designed two baseline approaches to solving the exam, one based on Information Retrieval and the other based on Word Embedding. We leave the details of these approaches to Section III.

Davis proposed creating a curated dataset called SQUABU (Science Questions Appraising Basic Understanding)a containing multiple-choice science questions taken from both High and Primary School Science Tests [8]. In this dataset the questions should assess understanding of time, causality, the human body and scientific methodology, including interpreting real world phenomena and laboratory experiments. This dataset (as of the time of this writing) has not been created.

Clark et al. [9] described a system called Aristo to solve 4th grade Fundamental School Science questions. The system combines five different solvers with different characteristics: one solver employs information retrieval techniques, two solvers build on statistical association between words and two solvers operate by deriving facts from a knowledge base. Individually, each solver obtains between 40% and 60% accuracy. Aristo combines them in an ensemble, achieving 70% accuracy in the test-set containing 129 questions. In a recent work [10] the same authors show a new dataset of science questions, of four candidate-answers, having this one about 7800 questions divided into two groups, "Easy" and "Challenging". They do not apply Aristo on this dataset; instead they use two solvers, achieving 20.26% and 26.97% accuracy in Challenging group. This suggests that Aristo generalizes poorly.

III. Previous Solutions

In this section we review our previously proposed approaches to solving the ENEM, based on Information Retrieval (IR) and on Word2Vec [3].

The first approach reformulates the problem of solving multiple-choice question as an information retrieval problem, an approach also adopted by Clark et al. [9] and Cheng et al. [5]. The heart of the approach is to attribute to each candidate answer a score that represents the likelihood, according to a knowledge base of text documents, of observing (verbatim) that answer and the question texts in a same document.

The queries are composed by the conjunction of the question statement and the text of a candidate answer. The conjunctive query ensures that the documents retrieved have at least some words of the statement and some of the candidate answer. The answer is selected according to the score, which is treated as a confidence measure in that answer. Different techniques can be obtained by varying the knowledge based used to derive the score. We investigated three knowledge base construction methods:

- Header Only (IR-H), where the KB is created dynamically for each question, and contains only the question header;
- Wikipedia (IR-W), where the KB contains articles extracted from the Brazilian Wikipedia;
- ENEM (IR-E), where the KB contains document composed of the concatenation of header, statement and correct answer extracted from questions of other ENEM exams (i.e., excluding the one being solved).

We also proposed two heuristics for combining the previous approaches:

- Non-Deciding Heuristic (NDH), which uses IR-H when its score is positive, otherwise uses either Wikipedia (called NDH-W) or ENEM (called NDH-E).
- Adding Heuristic (AH), which adds the confidence measures given by IR-H and either IR-H (called AH-W) or ENEM (called AH-E).

One of the drawbacks of IR-based approaches is to consider texts verbatim, that is to ignore the semantic similarity between phrases that do not share words. Word Embedding (WE) is a family of methods that learn vector representations of words such that semantically similar words are represented by similar vectors [11]. One of the most popular classes of Word Embeddings is the Word2Vec family, which uses neural networks.

We used Word2Vec vectors in two manners: to answer questions directly, and to augment IR queries by adding related words according to the WE model. Questions can be directly solved by comparing the distance between a vector V_a representing the answer and a vector V_q representing the question. V_a is obtained as the sum of the respective vectors of the words appearing in the candidate answer text; V_q is formed by adding the vectors of the words in either the statement or the header. The answer whose vector V_a maximizes the cosine similarity w.r.t. V_q is selected. We call this approach the *R1* heuristic. Query augmentation was performed as follows: for each word w_t in a query $q = w_1 w_2 \cdots w_N$ we added the word w_α which maximizes the cosine similarity w.r.t. w_t according to the WE model.

IV. Improvements

In this section we describe several improvements that we have made to our previously developed heuristics.

Concerning the IR-E heuristic, we have extended the size of its knowledge base by incorporating questions from a larger set of ENEM exams (from 2009 to 2017). The IR-W heuristics previously used a knowledge base whose documents were sentences extracted from Wikipedia articles. We modified this knowledge base so that each document now corresponds to (the text of) an article (obtained using a dump from January 2018). Concerning WE heuristics, we have previously used vector representations obtained by training Word2Vec networks on the Wikipedia corpus (the same used for IR-W then). We have now use pre-trained vectors trained on a large Portuguese corpora composed of Wikipedia articles, news, movie subtitles, etc [12]. We now also experimented with other types of neural network based WE, namely, Wang2Vec [13], FastText [14] and GloVe [15], and with different WE dimensions (50, 100, 300, 600, 1000); each type of WE arguably better captures different types of linguistic information, such as morphology, syntactic information and semantic information. As before, these WE were used either to augment queries in IR approaches, or to directly answer questions by comparing query and answer vector similarities.

We also investigated how to enhance IR heuristics by augmenting the entire text of the questions (i.e., the knowledge base) and not only the queries. This benefits the WE approach and the IR-H database while keeping the same advantages for the queries. To augment text we use the WordNet [6] (a lexical ontology where words are connected by relations such as synonymy, hypernymy, hiponymy and holonymy) and the GloVe WE model of size 300; the latter was used because it was the overall winner in a recently conducted analogy task in Brazilian Portuguese [12].

To augment text we create first a preprocessed version of the text, in which we transform all words to lowercase, eliminate punctuation marks and stop-words (we used Python NLTK stop-word list). From this preprocessed version we take all the remaining words w_t and add all the M words $w_{\alpha 1} ... w_{\alpha M}$ which have a particular relation r with w_t. Using WordNet we have access to the relations Synonym, Hypernym, Hyponym and Holonym, that is, for instance, we can augment text by adding the synonyms of every word of the text. Importantly, the Portuguese WordNet is not created originally in Portuguese, it translates a term to English and uses the English WordNet to get the relations, this introduces inherently some noise in the application. To augment using the WE model we define a "close words" relation, that is, all the words that, according to the model, have a cosine similarity higher than 0.5 with the word w_t. Additionally, we keep only the ten highest scoring close words. We set ten close words because most of the words have about five close words, while some few words have more than twenty close words and we did not want these few words to contribute much more than the others in the augmentation.

Consequently we have, for any Solver, 7 variations: Normal (i.e., without text augmentation), Preprocessed, Synonym, Hypernym, Hiponym, Holonym and (augmented by) WE. To simplify matters we use, for instance, IR-H-Synonym to designate the heuristic that uses the Information Retrieval approach with the Header Only database with text augmented using WordNet synonyms, or Wang-cbow-300-r2-Normal, to designate that we are using the Wang2Vec family, the CBOW model with vectors of size 300, using the R2 heuristic over the Normal text.

We highlight that the IR-E database was created using the Normal variation and we do not have variations of it using text augmentation, and in IR-H the text is augmented before indexing.

V. Ensemble

A popular approach to boost the predictive performance is to build ensemble of weakly correlated predictors. If we consider as an algorithm every variation of the IR and WE approach, then we have 539 different algorithms to solve ENEM questions. We would like to combine some of these algorithms in a way that each algorithms evaluates different aspects of the same question, and thus produces a combined answer that is more accurate and robust than individual answers. Similar approavhes have been used in Aristo [9] and in IBM's Watson QA system [16].

To combine the algorithms we find an ensemble by Beam Search guided by cross-validated accuracy where the final

```
1: procedure SEARCH(Algorithms, Beam_width)
2:     State.members ← ∅
3:     State.eval ← 0
4:     L ← State
5:     while L ≠ ∅ do
6:         Aux ← ∅
7:         for s ∈ L do
8:             if s.eval > Best.eval then
9:                 Best ← s
10:            Aux ← Aux ∪ (TopDown(s, Algorithms) ∪
    BottomUp(s) ∪ Alteration(s, Algorithms))
11:        Aux ← SortByEval(Aux)
12:        L ← Fetch(Aux, Beam_width)
13:    return Best
```

Fig. 1. Greedy Search Pseudo-code

answer is selected by the Majority Vote of its components. The pseudo-code of the Search algorithm is presented in Figure 1. A State (representing an ensemble) can expand in three different ways: by adding one solver to the ensemble (TopDown), by removing one solver from the ensemble (BottomUp) and by replacing one algorithm of the ensemble by another solver (Alteration). These three operations return only the new States that have better evaluation than the current State. Evaluation here is measured by the number of points achieved in solving the exams. Lastly, the returned states are ordered by their evaluation and we keep the first $Beam\_width$ States, repeating the process until no next State is found.

Alternatively, we used WE to answer questions indirectly, in the spirit of [9], [17]. This is done by extracting two features, and then using a SVM to classify the candidate answer as either true or false. The features extracted are the average of the cosine similarity of each word of the alternative with each word of the question, and the cosine similarity of V_q with V_a, as defined for *R2*. We used all the 35 vectors to extract these features, consequently, we have 70 features per candidate answer.

VI. Empirical Results

Dataset: We selected from the ENEM dataset only the questions not labeled as IC, CE or MR. That is, questions that do not require understanding images and chemical elements or solving mathematical problems. This restriction makes available 920 questions comprising 10 exams – one for each year from 2009 to 2017, having two exams in 2016.[1]

Evaluation: Algorithms were evaluated by the average of their accuracy when solving the exams. For every question the algorithms must return a list L of possible answers; if the correct answer is in L then the algorithm receives $\frac{1}{|L|}$ point (where $|L|$ is the size of L), otherwise the algorithm receives 0 points. This way an algorithm that answers all the options for every question obtains 20% accuracy and 0 standard deviation.

[1]The dataset is available at https://www.ime.usp.br/~ddm/enem/

TABLE I
COMPARISON BETWEEN THE PERFORMANCE OF THE IR SOLVERS. DISPLAYING PERCENTAGE OF AVERAGE FOLLOWED BY STANDARD DEVIATION.

Solver	Without augmentation	With augmentation	Variation
IR-H	20.9±2.5	24.1±3.4	Hyper
IR-E	23.3±4.7	25.0±3.8	Hyper
IR-W	26.9±3.8	26.4±3.8	Pre
NDH-E	23.0±3.9	25.2±4.1	Hyper
NDH-W	23.4±3.2	24.7±3.6	Hyper
AH-E	23.8±4.8	24.7±3.6	Hyper
AH-W	26.6±4.0	25.8±4.4	Pre

Training: The SVM and the Ensemble methods were trained using tenfold-like cross-validation, where one exam is used for testing and the remaining nine exams are used for estimating parameters or building knowledge bases. This gives us in average 835 training examples and 85 testing samples (but since we do not use all questions, the number of questions varies from exam to exam). We tune the SVM's soft margin parameter by an extra cross-validation within the training set. We used Beam Width of ten to find ensembles, to remain within reasonable training times (e.g., using Beam-width of 25 took about one week).

We start by presenting in Table I the difference in performances of the IR Solvers with (second column) and without (third column) text augmentation. We consider "without augmentation" the Normal variation and "with augmentation" all the others, Preprocessed included, even though it does not increase the size of the text. To save space we show only the result of the highest scoring augmentation and show in the last column which variation achieved that score.

The first thing to notice is that the 7 solvers even without augmentation are on average better than random-guessing. Still, as the standard deviation suggests, IR-H, IR-E, NDH-E and AH-E occasionally perform worse than a random guesser, while NDH-W very seldom performs worse than random. Secondly, when using augmentation the average of five solvers increase and two decrease (AH-W and IR-W), but all of them remain better than random, even when subtracting the standard deviation. Finally, we see the dominance of Hypernyms in achieving the highest scoring: only IR-W and AH-W are better in the Normal variation; in both approaches the best option is to change the text the least possible.

Small KB tend to have knowledge stated in more generic form, while questions are usually about particular cases, thus by using Hypernym we increase the chance of finding a relevant document, while, on the other hand, the augmentation tends to increase the size of the text, which may harm the performance of larger databases such as the Wikipedia.

In Table II we present the best performance of the four families of WE used, dividing them by model (CBOW and Skip), heuristic (R1 and R2), and with (w) or without (wo) text augmentation. As GloVe does not have models, we show it as CBOW.

In the *W2V* family it is always better to augment text and use R2 instead of R1, being the highest achievers R2

TABLE II
COMPARISON BETWEEN BEST PERFORMANCE OF WORD EMBEDDING METHODS. DISPLAYING ACCURACY AND STANDARD DEVIATION IN PERCENTAGE.

Heuristic	Word2Vec	FastText	Wang2Vec	GloVe
CBOW R1 (w)	25.7±3.7	26.9±5.0	25.4±3.9	23.5±4.3
CBOW R1 (wo)	23.9±3.9	23.8±2.7	24.7±4.5	21.9±3.8
CBOW R2 (w)	27.1±2.4	26.0±4.0	25.0±3.0	25.0±3.8
CBOW R2 (wo)	25.0±2.2	23.1±2.6	25.9±4.6	24.5±3.8
SKIP R1 (w)	25.3±3.5	25.4±6.9	24.4±5.0	—
SKIP R1 (wo)	24.7±4.5	24.3±4.9	22.9±3.0	—
SKIP R2 (w)	26.7±4.3	28.0±4.5	24.8±4.0	—
SKIP R2 (wo)	25.7±3.3	27.1±5.6	24.4±5.1	—

with augmentation, CBOW's best variation was Hypernym and Skip's best variation was Hypernym, both of embedding size of 600. *FastText* also benefits from text augmentation, but in its CBOW model the R1 heuristic is better than R2, while in the Skip model R2 remains better than R1. The highest scoring augmentations for FastText are: Cbow-R1-Synonym, Cbow-R2-Hypernym, Skip-R1-Hypernym, Skip-R2-Holonyms. Five out of the eight FastText's performances presented come from embedding of size 1000. *Wang2Vec* has one case in which it is better not to enhance text; this happens in CBOW-R2, but by augmenting the text the accuracy drops 0.9% and the standard deviation drops 1.6%. In the Skip model of Wang2Vec the R2 heuristic performs better than R1. The best augmentations were Hyperonym for Wang-Cbow-R1 and Wang-Skip-R2, Synonym for Wang-Cbow-R2 and Hyponym for Wang-Skip-R1; out of the eight performances displayed, three are of 1000 dimensions embedding, two of 300, one of 600, 100 and 50. Finally, *GloVe* also benefits from text augmentation and R2 has better accuracy than R1. The winning variations were Holonym (R2) and Hyponym (R1); out of the 4 performances displayed, half are from 300 size embedding, one of 1000 and 600.

In short, higher dimensional vectors have better performance, R2 has advantage over R1, Hypernym here also appears frequently as the best augmentation and using augmentation always has an advantage. Considering only the best R2 performance, CBOW wins over Skip two times.

As *R2* tends to be better than *R1*, to extract features we used V_q as defined by *R2* and used only the Normal variations. We also tested the four features presented in [18], a follow-up paper that presented the two features used by Aristo's SVM, but the results were not significantly different, so we omit them here. We used the SVM implementation of scikit for Python. The results for Linear, Rbf, Poly (degree three) and Sigmoid kernels are shown in Table III.

The Linear and the Polynomial kernels achieved the best performances, 26.6% and 25.3%, respectively. A competitive result when compared to the performances without augmentation of the previous approaches. In fact, the performance of the Linear kernel is only the third highest performance without augmentation, losing to IR-W-Normal (26.9%) and FastText-Skip-R2 (27.1%). On the other hand, it obtains the smallest standard deviation (viz. 3.2%), while the IR one has 3.8 and the WE one has 5.6. Rbf is averagely better than random, but its deviation shows that it is not always the case, the same happens with the Sigmoid kernel, but its deviation is higher (6.2) and its accuracy is almost random (20.8%).

TABLE III
PERFORMANCE OF THE FOUR KERNELS OF SVM. ACCURACY IS PRESENTED IN PERCENTAGE.

Metric\Kernel	Linear	Rbf	Poly	Sigmoid
Accuracy	26.6	23.6	25.3	20.8
Std. Deviation	3.2	4.3	3.0	6.2

TABLE IV
PERFORMANCE OF THE ENSEMBLE PRESENTED IN PERCENTAGE.

year	train	test	size	iterations
2009	36.95	26.96	28(26)	32
2010	35.68	31.37	18(17)	19
2011	35.13	33.85	14(14)	15
2012	35.13	31.34	19(18)	21
2013	34.65	30.33	20(19)	21
2014	36.21	27.01	23(22)	25
2015	33.65	29.96	09(09)	10
2016	36.02	28.42	23(22)	27
2016(2)	36.00	27.95	14(14)	14
2017	33.96	26.68	09(09)	09
Avg:	35.33	29.39	17.69	—
Std. Dev.	0.98	2.25	5.9	—

The Ensemble's performance is presented in Table IV. The second column presents the accuracy in the training set, the third column presents the accuracy in the test set, the size column represents how many algorithms compose the final ensemble and in brackets how many different algorithms there are in the ensemble, the last column presents how many iterations were necessary to compose the group.

It seems that the Ensemble matches its performance in the test set and training set. In six cases the method repeated at least one time an algorithm that was already in the ensemble; this can be seen as a type of learning, that is, the method learned to weight differently one of the algorithms. In all cases the ensemble was not composed by simply aggregating more solvers, meaning that the three functions that compose the Beam Search are relevant (otherwise the number of iterations would be equal to size). Compared to all the previous approaches, the Ensemble has the highest accuracy (29.3%) and the lowest standard deviation (2.2%).

We show in Table V the performance of the highest scoring algorithms of IR, WE, SVM and Ensemble per knowledge tag: 26.9% for IR-W-Normal, 28% for FastText-Skip-1000-Holonyms-R2, 26.6% for SVM-Linear, 29% for Ensemble. We treated the whole dataset as one exam, that is, without differentiating between years. Note that there are only 18 questions tagged solely as Domain Specific.

As expected of Information Retrieval, its best performance is on questions that rely solely on EK and its worst performance is on questions that rely exclusively on DS; in these cases the algorithm performs worse than random. Word Embedding also fulfills the expected: its better performance is on purely TC questions, and in all tags its performance

TABLE V
COMPARISON OF PERFORMANCE BASED ON QUESTION TAGS. THE HIGHEST SCORING ALGORITHMS OF EACH APPROACH WERE SELECTED. SCORES ARE IN PERCENTAGE.

	TC	EK	DS	TC_{only}	EK_{only}	DS_{only}
#Questions	778	411	176	402	98	18
IR	26.4	28.4	25.5	25.8	33.6	16.6
WE	28.7	27.4	28.4	29.8	23.4	27.7
SVM	26.9	25.5	23.8	28.6	27.5	16.6
Ensemble	32.5	32.3	28.9	33.5	37.7	16.6

is better than random, this is the only algorithm (of the four shown in the table) that is better than random on DS only questions. SVM seems like a weaker version of WE, but its performance on questions of pure EK is better than that of WE. The Ensemble has the best performance on all tags but one, achieving about 30% in all but one; its best performance is on EK only questions (37.7%) and the worst is on DS only questions (16%). From this last table we see that IR and WE are complementary in their strengths, SVM without augmentation is competitive with the other approaches, the Ensemble surpasses the best performance of its components, and DS only questions are particularly challenging.

VII. CONCLUSION

This paper follows up and improves on our previous work on designing baseline heuristics for Multiple Choice Question Answering using questions from the Brazilian university entrance exam Exame Nacional do Ensino Médio (ENEM). Our previous solutions formulated the problem of answering a ENEM question as an Information Retrieval (IR) problem where each candidate answer is assigned a score representing its similarity to the statement. We also experimented with word embedding approaches that produce a vector representation of statement and candidate answer text; the answer with the smallest distance is then selected.

We improve our previous solutions by incorporating more questions to the knowledge base of IR approaches, by augmenting IR queries and knowledge bases with similar words, by using word embeddings trained on larger corpora of Portuguese text, and by considering ensembles of weakly correlated solvers. For the text augmentation part we investigate both augmentation by word embeddings and by relations extracted from WordNet.

We observe that IR-based solvers benefit from text augmentation when the database is relatively small. In this case the best scoring algorithm achieves 25.2% when augmenting text with hypernym relations from WordNet. For large databases the text augmentation hurts performance.

We developed a new heuristic to use with word embeddings, which combined with text augmentation techniques obtains 28% accuracy (29.8% accuracy on questions requiring text comprehension). We also developed a SVM solver that uses features extracted from different word embeddings of question and candidate answer texts and achieves 26.6% accuracy (without augmentation). As with word embedding, the best performance of this solver is in Text Comprehension (28.6%).

Finally, we showed that by a greedy search it is possible to create an ensemble of algorithms that capture different aspects of a question. This Ensemble achieved 29% accuracy, having as weakness questions of Domain Specific knowledge, its best performance was 37.7% on pure Encyclopedic Knowledge questions.

REFERENCES

[1] P. Brasil, "Mais de 9,2 milhões se inscreveram no ENEM 2016," (http://www.brasil.gov.br/educacao/2016/05/mais-de-9-2-milhoes-de-candidatos-se-inscreveram-no-enem), 2016, [Online; accessed April-29-2018].

[2] T. Yuan, "Number of gaokao students dwindling overall," (http://english.cctv.com/2016/06/07/VIDElr9mDklWGIfFywe1zPLK160607.shtml), 2016, [Online; accessed April-29-2018].

[3] I. C. Silveira and D. D. Mauá, "University entrance exam as a guiding test for artificial intelligence," in *6th Brazilian Conference on Intelligent Systems*, 2017.

[4] Y. Miyao and A. Kawazoe, "University entrance examinations as a benchmark resource for NLP-based problem solving," in *Proceedings International Joint Conference on Natural Language Processing*, 2013, pp. 1357–1365.

[5] G. Cheng, W. Zhu, Z. Wang, J. Chen, and Y. Qu, "Taking up the gaokao challenge: An information retrieval approach," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, ser. IJCAI'16. AAAI Press, 2016, pp. 2479–2485.

[6] G. A. Miller, "Wordnet: A lexical database for english," *Communications of the ACM*, vol. 38, pp. 39–41, 1995.

[7] A. Fujita, A. Kameda, A. Kawazoe, and Y. Miyao, "Overview of Todai Robot Project and evaluation framework of its NLP-based problem solving," in *Proceedings of the 9th International Conference on Language Resources and Evaluation*, 2014.

[8] E. Davis, "How to write science questions that are easy for people and hard for computers," *AI Magazine*, vol. 37, pp. 13–22, 2016.

[9] P. Clark, O. Etzioni, T. Khot, A. Sabharwal, O. Tafjord, P. D. Turney, and D. Khashabi, "Combining retrieval, statistics, and inference to answer elementary science questions," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016, pp. 2580–2586.

[10] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, "Think you have solved question answering? Try ARC, the AI2 reasoning challenge," *Unpublished*, 2018.

[11] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems*, 2013, pp. 3111–3119.

[12] N. Hartmann, E. R. Fonseca, C. Shulby, M. V. Treviso, J. Silva, and S. M. Aluísio, "Portuguese word embeddings: Evaluating on word analogies and natural language tasks," in *STIL*. Sociedade Brasileira de Computação, 2017, pp. 122–131.

[13] W. Ling, C. Dyer, A. W. Black, and I. Trancoso, "Two/too simple adaptations of word2vec for syntax problems," in *HLT-NAACL*. The Association for Computational Linguistics, 2015, pp. 1299–1304.

[14] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[15] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation." in *EMNLP*, vol. 14, 2014, pp. 1532–1543.

[16] D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager, N. Schlaefer, and C. Welty, "Building Watson: An overview of the DeepQA Project," *AI Magazine*, vol. 31, 2010.

[17] P. Jansen, M. Surdeanu, and P. Clark, "Discourse complements lexical semantics for non-factoid answer reranking," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2014, pp. 977–986.

[18] D. Fried, P. Jansen, G. Hahn-Powell, M. Surdeanu, and P. Clark, "Higher-order lexical semantic models for non-factoid answer reranking," *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 197–210, 2015.